Chapter 5

Packet Scheduling Principles and Algorithms for Downlink

Kamal Deep Singh, Gerardo Rubino, and César Viho

Contents

5.1 Introduction

This chapter presents the packet scheduling principles and different scheduling algorithms for High-Speed Downlink Packet Access (HSDPA). HSDPA is an enhancement of UMTS (Universal Mobile Telecommunications

System) networks that supports data rates of several megabits per second (Mbps), making it suitable for data applications ranging from file transfer to multimedia streaming. Despite the fairly high data rates that HSDPA offers, the shared downlink radio channel used in HSDPA is a challenging environment for delay- and loss-sensitive applications like video streaming. Thus, resource management algorithms are needed to ensure good quality-of-service.

MAC (mediam access control) layer packet scheduling is one of the salient points of HSDPA to perform resource management (i.e., bandwidth allocation between terminals), taking into account the radio channel conditions of all users. In some proposals, additional factors such as fairness between users or cell throughput or quality-of-service (QoS) parameters are also considered in the scheduling mechanism.

The focus of this chapter is on downlink-centric services in the 3G network. For this reason, other services such as VoIP (Voice over Internet Protocol), which are also dependent on uplink, are not discussed in detail, although some discussion about them is provided at the end. Moreover, the services considered in this chapter, such as video streaming, have relatively flexible delay requirements as compared to VoIP. In addition, it should be noted that the emphasis of this chapter is only on MAC layer scheduling and other UTRAN (UMTS Terrestrial Radio Access Network) functions, shown in Figure 5.1, that affect the QoS are out of scope of this chapter.

This chapter is structured as follows. First the key concepts related to HSDPA scheduling are presented. Then, different packet scheduling

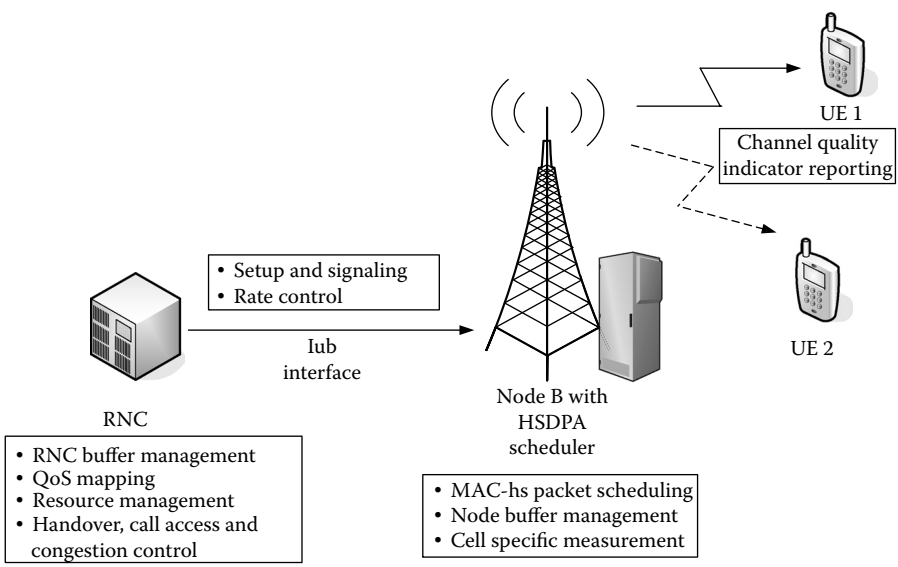

Figure 5.1 Overview of functional split between RNC, Node B, and UE.

algorithms are discussed and their performance analyses is done. The chapter concludes after a short discussion of some points that are not considered in this chapter.

5.2 Packet Scheduling

The HSDPA scheduler is the key to resource management in the UTRAN downlink because it decides which user is to be scheduled in each transmission time interval (TTI). In general, the scheduling decision does not take into account only the channel conditions, but additional factors such as fairness between users, cell throughput, and QoS parameters are typically considered in the scheduling mechanism. The choice of a scheduler usually involves some trade-offs between these factors.

In HSDPA, a new transport channel called high-speed downlink shared channel (HS-DSCH) has been introduced. HS-DSCH is supported by an auxiliary channel called high-speed shared control channel (HS-SCCH). Node B uses HS-SCCH to send the parameters related to demodulation and decoding and sends this information two slots in advance. The value of TTI in HSDPA has been reduced to 2 ms, which is equal to three slots, as compared to the 10, 20, 40, and 80-ms intervals supported by UMTS Release 99. In the uplink, high speed dedicated physical control channel (HS-DPCCH) is used to feed back ACK/NACK and to allow for fast monitoring of the radio channel conditions of all users: Every TTI of 2 ms, a UE (User Equipment) can send a channel quality indicator (CQI) to Node B over this control channel. The relationship between these channels is shown in Figure 5.2.

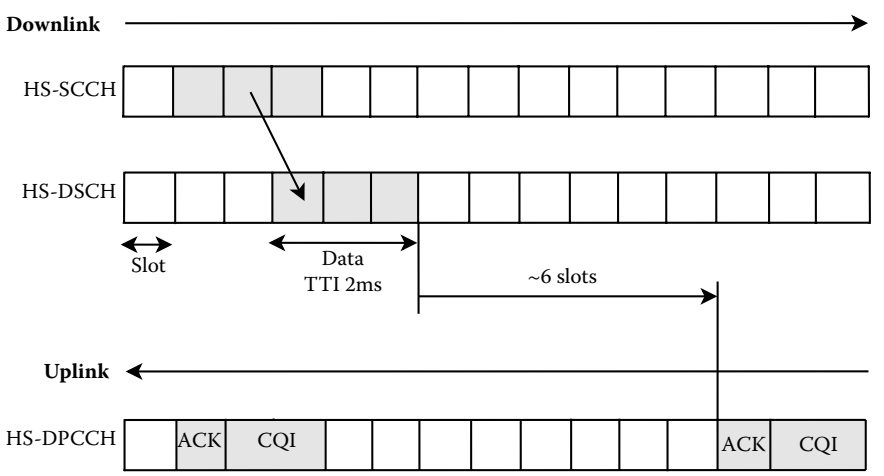

Figure 5.2 HSDPA channels and timing relationship.

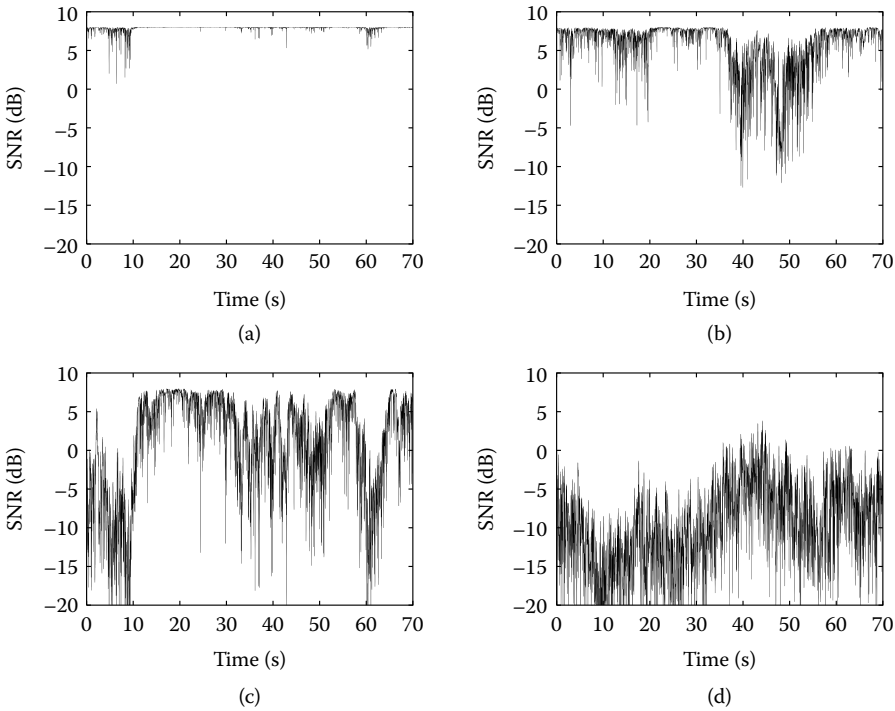

Figure 5.3 Example of received power by UEs (Ped A, 3 km/h) at different distances from Node B: (a) 50 m, (b) 200 m, (c) 350 m, and (d) 500 m.

CQI feedback makes it possible to optimize the transmission by means of channel-adaptive schemes. The quality of reception can vary due to variable radio channel conditions over time. There are many factors such as distance of UE from Node B, transmission power, fading, noise that can affect the reception quality. For example, in Figure 5.3 it can be seen that the channel conditions,[1] especially for the farther users, may change rapidly. To improve the chances of correct reception, the robustness of the transmitted signal is adapted to the channel conditions and this is called Adaptation, Modulation and Coding (AMC).

With AMC, HSDPA can adaptively choose the modulation and coding for a given time slot as a function of current channel quality. While choosing modulation and coding, the trade-off is between robustness and data rate. This approach, combined with the packet scheduling and CQI, has certain advantages. For instance, the users experiencing good channel quality can

[1] The curves in the figure were obtained through the simulation [1] of a PHY-layer model.

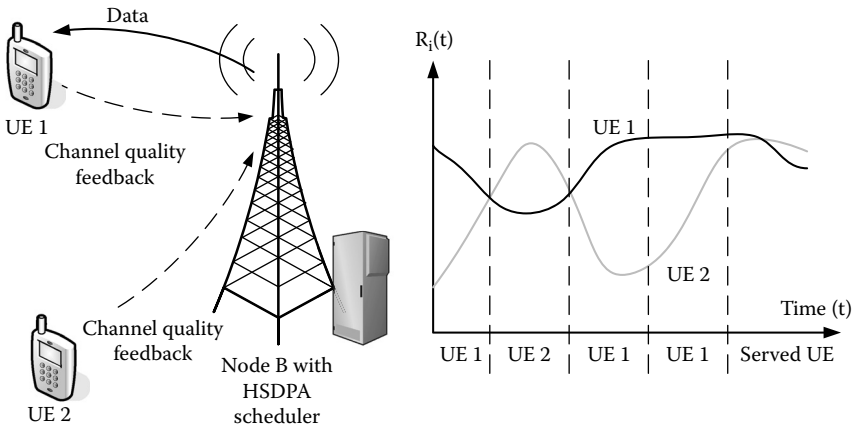

Figure 5.4 Example of channel-aware HSDPA scheduling. Here the user with the best channel conditions, in the given TTI, is scheduled.

benefit from high data rate and, in the presence of several users, the packet scheduling algorithms can benefit from user diversity and, thus, schedule users during constructive fades. In the case of several users, the resource management is performed by the scheduler that may adapt to fluctuating radio conditions. Every TTI, the scheduler chooses the next user to be served based on the channel conditions of *all* the users. This channel-adaptive scheduling can be used, for instance, to maximize the global cell throughput by scheduling the user with best channel condition, as shown in Figure 5.4. Note that a given UE can experience strong fluctuations in bandwidth over time, due to such channel-dependent scheduling policies.

5.2.1 Quality-of-Service-Aware Scheduling

Some applications, such as VoIP or multimedia streaming, have requirements that cannot be satisfied with *Best-Effort* service when there are not enough network resources available. This can occur when the network is congested or when other services, those without stringent requirements, compete aggressively for available resources. For these applications, the network infrastructure must be adapted to give them special treatment in terms of minimum bandwidth, delay, loss tolerance, jitter, etc. This special treatment makes it possible to satisfy the specific requirements of different services and ensures good QoS.

Different QoS parameters or UMTS bearer attributes such as traffic class and traffic handling priority are defined in [2]. However, the values of these parameters are not available at Node B where the HSDPA scheduler is implemented. In fact, Node B can also be from a manufacturer that is

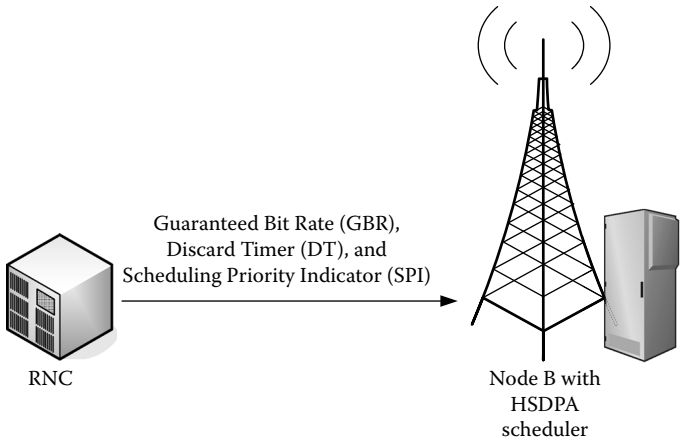

Figure 5.5 RNC sets QoS parameters for HSDPA scheduler present in Node B.

different from that of the radio network controller (RNC). Thus, an interface between the RNC and Node B is standardized and the RNC is responsible for setting QoS parameters in Node B through the Iub interface [3], as shown in Figure 5.5. Some of the relevant parameters are guaranteed bit rate (GBR), the minimum guaranteed rate at which the user data will be scheduled; scheduling priority indicator (SPI), the priority with respect to other flows or users; and discard timer (DT) [3], the value in seconds or milliseconds such that any packet delayed more than this value must be discarded by Node B. Note that the RNC has to map all the QoS parameters defined in Release 99 to these parameters in order to provide QoS.

After these parameters are configured by the RNC, it is then up to the HSDPA packet scheduler algorithm to use these values for serving different users. 3GPP does not specify the exact behavior of the scheduler as a function to these parameters and, hence, this is kept open for the implementation. The choice of values for these parameters or mapping of different UMTS QoS class attributes to these parameters by the RNC is also operator dependent. In addition, the operator must decide how to handle the user traffic when it exceeds GBR or when the user data rate is variable, even when the average bit rate remains equal to the GBR.

In some cases, the RNC need not set the DT at Node B. For example, in the case of multimedia streaming, where delays on the order of some seconds are tolerated, it may disable the DT at Node B. Nevertheless, it can still use a DT at the RNC such that, on timeout, it can drop full IP packets before segmenting them into RLC-PDUs (Protocol Data Units) and before sending the segments to Node B. On the other hand, when VoIP service is used, the delay requirements are very strict and are on the order of 100 ms. In this case, the RNC should avoid queuing the IP packets and instead should send

them to Node B as soon as possible. In addition, it should configure the DT at Node B, (e.g., to a value of 100 ms) to avoid queue buildup at Node B. That way, any queue buildup will be avoided because Node-B queues will always be emptied after timeout because either the enqueued data will be transmitted or it will be discarded. This will work even when a flow control algorithm is used between the RNC and Node B, because after the queue is emptied at Node B, it will automatically ask for more data from the RNC.

Regarding the decision to set SPI values, usually the higher priority will be set for VoIP flows, lower priority for streaming flows, and lowest priority for the flows belonging to non-real-time traffic.

5.3 Packet Scheduling Algorithms

Numerous MAC-layer schedulers have been proposed and studied in the literature. Here we discuss some of the schedulers that have been proposed for HSDPA and are representative of different design trade-offs and constraints. We divide the set of schedulers into two parts and call them Best-Effort schedulers and QoS schedulers. Best-Effort schedulers do not provide QoS guarantees, whereas QoS schedulers can provide QoS guarantees. Further explanation about them is given in the following text.

The simulations in this section are done using a packet-level simulator. To simulate HSDPA, this chapter uses the Enhanced UMTS Radio Access Network Extensions for ns-2 (EURANE) extensions [1] to ns-2 [4]. The simulation settings for UTRAN are as follows. The Okamura-Hata distance loss model and correlation model [1,5] for shadow fading are used with 10 W of Node-B transmission power resources for HS-PDSCH, 17 dBi of antenna gain, 30 dBm of intra-cell interference, and −70 dBm of inter-cell interference. UM mode is used in the RLC layer, UE category 1-6 is assumed, and thus the maximum number of available codes for HS-DSCH is 5 and the maximum instantaneous bit rate is 3.6 Mbps.

The multipath environment used is Pedestrian-A with a user speed of 3 kmph. The cell radius is assumed to be 500 m. In the simulations, the distance of users from Node B is randomly picked to be $500\sqrt{u}$ meters, where u is a uniformly distributed variable between 0 and 1.0. Note that this approach ensures uniform distribution over a circle of radius 500 m, whereas directly picking the values of user distance equal to a uniform random variable between 0 and 500 m, instead, does not ensure uniform distribution. Moreover, once the distance is chosen for a user, it is assumed that the user moves in a circle without changing the distance from Node B to keep the simulations tractable with respect to user distance. The simulations are run 20 times for each simulation configuration to obtain good confidence intervals. The parameters that change are explicitly specified in the following text, and other detailed parameters are the same as the default settings used in [1].

5.3.1 Best-Effort Schedulers

Best-Effort schedulers do not provide any QoS guarantees to different flows. Mapping of some flows to Best Effort can be done, for example, by mapping Interactive class and Background class flows to Best Effort. The QoS parameters, such as GBR are set to zero. Because Best-Effort flows do not have strict requirements and QoS guarantees are not provided to them, the main objective of the scheduler algorithm in this case is to optimize the global throughput and ensure that resources are divided fairly among different users. A summary of some Best-Effort schedulers is provided below. Please refer to Table 5.1 while reading it.

The Round Robin (RR) scheduler, which is probably the simplest one in terms of implementation, gives the time slot to users in a cyclic manner. Note that RR is fair with respect to system resources, or time slots, but in general the capacity is *not* shared equally among the UEs, as shown in Figure 5.6 that shows the cumulative distribution function (CDF) of user throughput when ten FTP users are in the system. This is explained by the fact that each user may experience different channel conditions. Moreover, RR policy is not optimal with respect to maximizing the global throughput, as is shown in Figure 5.7, because this policy does not utilize channel condition feedback.

The most aggressive policy to utilize the channel feedback is called Maximum C/I scheduling.[2] This is the same algorithm as illustrated in Figure 5.4. The Max C/I scheduler gives the channel to a user having the best channel conditions, which in turn is the user that can support the maximum instantaneous data rate in the current TTI. This algorithm is also summarized in Table 5.1. Note that $R_i(t)$, used in Table 5.1, is the instantaneous supportable data rate at which user i can be served at time t, depending on CQI and $BLER_{target}$, which is typically 10%.

The Max C/I scheduler provides the highest *cell* (global) throughput because it always serves the users that can support the highest data rates. This is shown in Figure 5.7. On the other hand, this scheme is very unfair because a user nearest to Node B can get all the resources, and the users farther away will be starved. This can be seen in Figure 5.6(a), which shows the CDF of user throughput. It can be seen that some users get very good throughput, up to 1500 kbps, because they are scheduled most of the time due to their good channel quality, whereas some other users get very bad throughput in the range of 10 kbps to 30 kbps because of their relatively poorer channel conditions.

Unlike the Max C/I (Carrier-to-Interference Ratio) scheduler, the Proportionally Fair (PF) scheduling algorithm [6–8] looks at the relative channel

[2] Max C/I denotes the signal-to-noise ratio that is used to estimate the optimal value of the instantaneous supportable data rate, $R_i(t)$, with a given block error rate target ($BLER_{\text{target}}$).

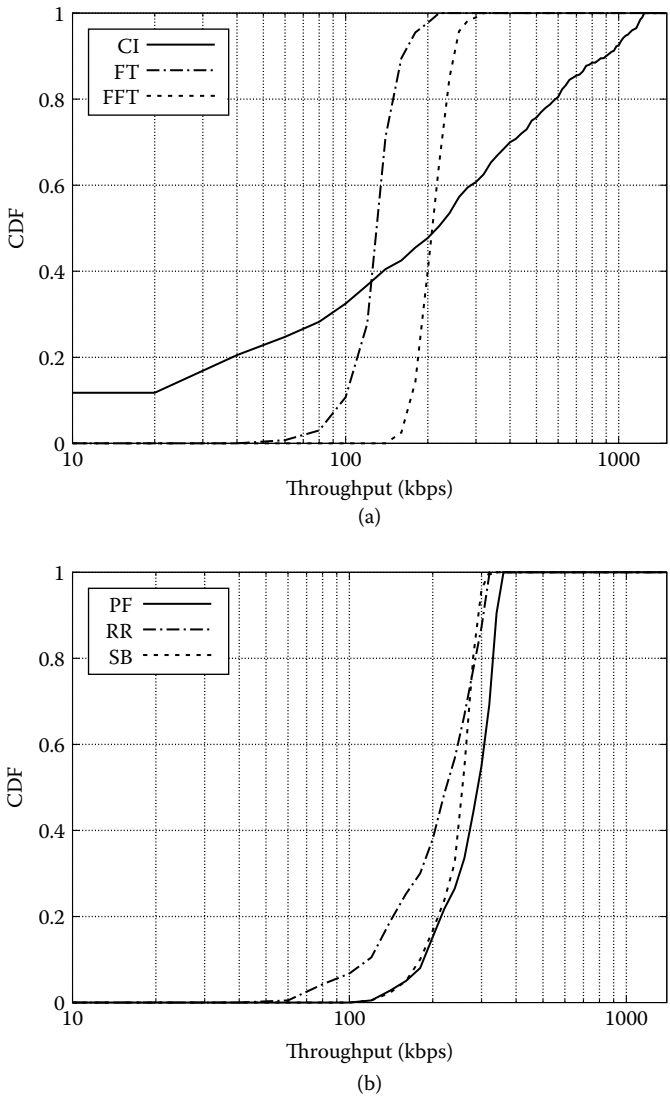

Figure 5.6 Comparison of different Best-Effort schedulers. (a) Comparison of CI, FT, and FFT schedulers and (b) comparison of PF, RR, and SB schedulers. CDF of user throughput over a time window of 1 minute is plotted. The bin size used for calculation of CDF is 20 kbps. There are ten FTP users in the system.

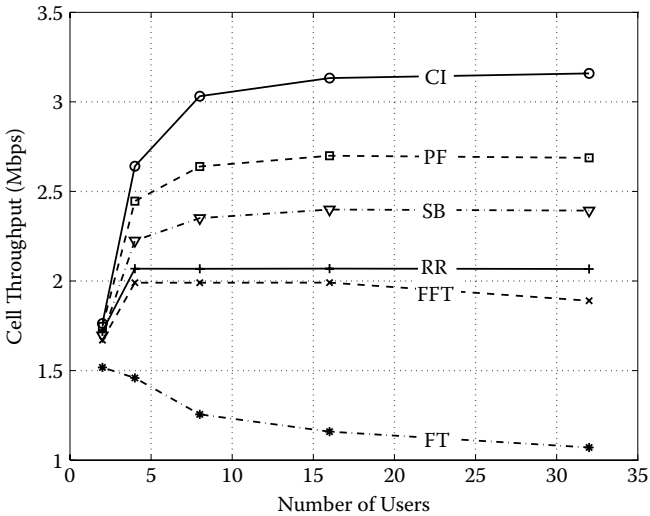

Figure 5.7 Cell throughput.

conditions of a user with respect to its own past channel conditions. PF assigns the current time slot at time t to a user i^* having maximum ratio of supportable instantaneous rate $R_i(t)$ and its own exponentially weighted moving average throughput $\lambda_i(t)$, which is calculated as follows:

$$\lambda_i(t) = \left(1 - \frac{1}{\tau}\right) \cdot \lambda_i(t - \Delta t) + \frac{s_i}{\tau} \cdot R_i(t) \tag{5.1}$$

with $\tau > 1$, Δt is equal to the length of the TTI, $s_i = 1$ when user i is served in the current time slot, and $s_i = 0$ otherwise. It should be noted that, with respect to Equation (5.1), different implementation variants are possible in order to deal with a few special cases such as when the user queue is empty or the data present in the queue of the scheduled user queue is less than that corresponding to $R_i(t)$. In the implementation considered in this chapter, when user i is being scheduled and the data in this queue is less than that corresponding to $R_i(t)$, we replace $R_i(t)$ in Equation (5.1) with the rate corresponding to the actual amount of data in the queue. Moreover, when a user's queue is empty, and thus is not being scheduled in the current time slot, $\lambda_i(t)$ is not updated and $\lambda_i(t) = \lambda_i(t - \Delta t)$.

The PF scheduler offers a good trade-off between cell throughput (see Figure 5.7) and fairness (see Figure 5.6(b)), as it gives the channel to the user having "relatively good" channel conditions and also results in a fair distribution of resources. This scheduler provides the so-called proportional fairness criterion defined in [9]. A Generic PF (GPF) scheduler [9,10] is also shown in Table 5.1. The GPF scheduler uses the ratio of $[R_i(t)]^\eta$ and $[\lambda_i(t)]^\gamma$

Table 5.1 Most Studied Best-Effort Scheduling algorithms

Scheduler	*Algorithm (Note that ties are broken randomly):*	*Scheduled User i* Is Given by:*
Round Robin (RR)	Schedule users in cyclic order	Cyclic manner
Fair Throughput (FT)	Equalize throughput of all users	$\arg\min_i \{\lambda_i(t)\}$
Max C/I	Choose user with best instantaneous supported data rate	$\arg\max_i \{R_i(t)\}$
Proportionally Fair (PF)	Choose user with best transmission rate relative to user's own average throughput	$\arg\max_i \left\{\frac{R_i(t)}{\lambda_i(t)}\right\}$
Generic Proportionally Fair (GPF)	Similar to PF algorithm with additional exponents γ and η	$\arg\max_i \left\{\frac{[R_i(t)]^\eta}{[\lambda_i(t)]^\gamma}\right\}$
Score Based (SB)	Choose user with best rank for channel quality based on history	$\arg\min_i \left\{S_i(t)\right\}$
Fast Fair Throughput (FFT)	Equalize throughput of all users and exploit channel conditions	$\arg\max_i \left\{\frac{R_i(t)}{\lambda_i(t)}\frac{max_j(\overline{R_j(t)})}{\overline{R_i(t)}}\right\}$

Note: Here, for a given user i at time t, $\lambda_i(t)$ is the exponentially weighted moving average throughput, $R_i(t)$ is the instantaneous supportable data rate, $\overline{R_i(t)}$ is the exponential weighted moving average of $R_i(t)$, $max_j(\overline{R_j(t)})$ is the maximum among different users, $S_i(t)$ is the rank of the current channel condition with respect to past W values for the same user. In addition, W, γ, η are implementation parameters.

for deciding the user to be scheduled. When the exponents γ and η are close to 1.0, then the GPF scheduler behaves similarly to the PF scheduler. Whereas, when γ is increased more than 1.0, or η is decreased less than 1.0, then GPF starts behaving in an "extra" fair manner by favoring users with bad channel conditions. On the other hand, when γ is decreased less than 1.0 or η is increased more than 1.0, then it starts behaving like the Max C/I scheduler. This can be seen in Figure 5.8 that shows the CDF of user througputs with different values of γ and η.

The work in [11] argues that the PF scheduler results in fair distribution only under ideal conditions, when users experience symmetric fading statistics, and becomes unfair in realistic conditions. A Score-Based (SB) scheduler is proposed that ranks the current channel condition of a user with respect to the past channel conditions over a window of size W. For example, the current and past W values $R_i(t)$ to $R_i(t-W+1)$ for a user i, are ranked from 1 to W, depending on their values. Similarly, the current

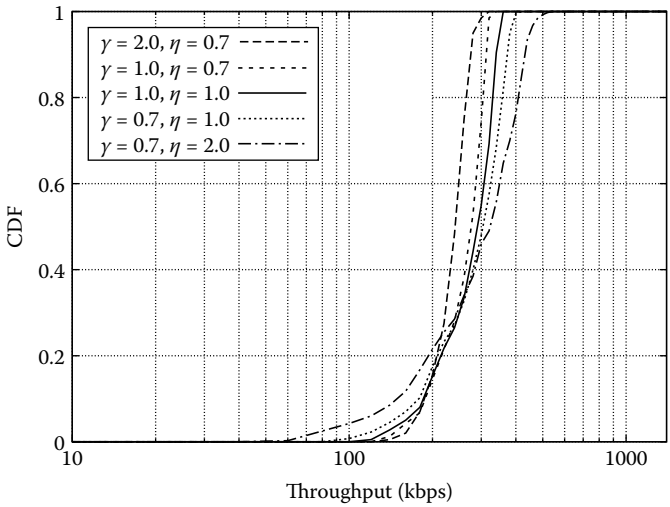

Figure 5.8 Generic PF scheduler with different values of γ and η.

channel conditions of other users are ranked based on their respective past channel conditions. After that, users are sorted according to the rank of their current channel condition and then the user having the best rank is scheduled in the current TTI. Note that ties are broken randomly and similarly, while in ranking, if two values related to channel conditions are the same for a given user, then either of them is ranked better than the other with equal probability. It is reported in [11] that SB scheduling is able to distribute the time slots equally even when asymmetric channel variations are encountered. Figure 5.6 shows the CDF of user throughput obtained with SB and from Figure 5.7 it can be seen that equal distribution of time slots is achieved at the cost of slightly decreased overall cell throughput when compared with the PF scheduler. However, the cell throughput obtained with SB is still higher than that obtained with RR as shown in Figure 5.7.

In fact, the implementation of SB scheduling used in this chapter is slightly different and better than in [11]. It is suggested in [11] that in the case of a tie, while comparing the current channel condition with one of the past values, either of them is ranked better than the other with equal probability and comparison is moved to next value in the window. However, during the simulations done for this chapter, it was found that this approach can lead to throughput starvation when most of the past values are the same. For example, when past W values are the same as the current CQI representing current channel conditions, then after W coin tosses, it is expected that the unbiased coin will produce one of the two outcomes $\frac{W}{2}$ times. Thus, the rank comes to be around $\frac{W}{2}$ most of the time. This is a relatively bad rank; and when the number of users in the system

increases, then the chances are high that at least one user will have a better rank than $\frac{W}{2}$. This means that the chances of a user having similar past W or numerous CQI values getting scheduled will be much less.

This particular case of a user having numerous similar past CQI values is more frequent with users close to Node B because they experience very good channel conditions and most of the time the CQI value will be the maximum value possible and thus will be same. In the implementation used in this chapter, instead of tossing an unbiased coin each time, when a value equal to the past value is encountered, the implementation proceeds as follows: Denoting by W_s the number of past CQI values in the window that are equal to current CQI, a random integer between 0 and W_s having a uniform distribution, is drawn and added to the rank of current CQI obtained after comparing it with the unequal past CQI values in the window. It was observed during the simulations, the results of which are shown in Figures 5.6(b) and 5.7, that this approach worked and the users near Node B were no longer starved.

There exist other schedulers that use different criteria for fairness. A scheduler called Fair Throughput (FT) equalizes the throughputs of all users by scheduling the user with minimum average throughput. With FT, all users in the system, at a given time, obtain equal throughput. In Figure 5.6(a), the CDF is not exactly vertical because although the throughputs are the same at a particular time, even then they can still vary from time to time. FT has a drawback in that it does not exploit channel condition feedback. Moreover, it should be noted that with the FT scheduler, some users experiencing bad channel quality can cause deterioration of throughput for other users as well. This leads to a low value of overall cell throughput, as shown in Figure 5.7. This is because the scheduler will start providing more resources to the users with bad quality, which in turn would mean low average throughput for all the users. Thus, if FT is used, then it will have to rely on resource management procedures to drop some users experiencing bad quality in order to avoid deteriorating the throughput for other users. Another version of the FT scheduler is Fast Fair Throughput (FFT). FFT, proposed in [12], aims to provide fair throughput distribution as well as exploit channel condition feedback. As shown in Table 5.1, this distribution is achieved by multiplying the term in PF by the ratio of $max_j(\overline{R_j(t)})$ to $\overline{R_i(t)}$, where $\overline{R_i(t)}$ is the exponential weighted moving average of $R_i(t)$ and $max_j(\overline{R_j(t)})$ is the maximum $\overline{R_j(t)}$ among different users.

5.3.2 QoS Schedulers

QoS guarantees will be required by users of services such as video streaming that have requirements in terms of minimum bit rate and maximum packet delay. User satisfaction is the main factor to consider. Satisfaction criteria are different for different services, each having its own QoS

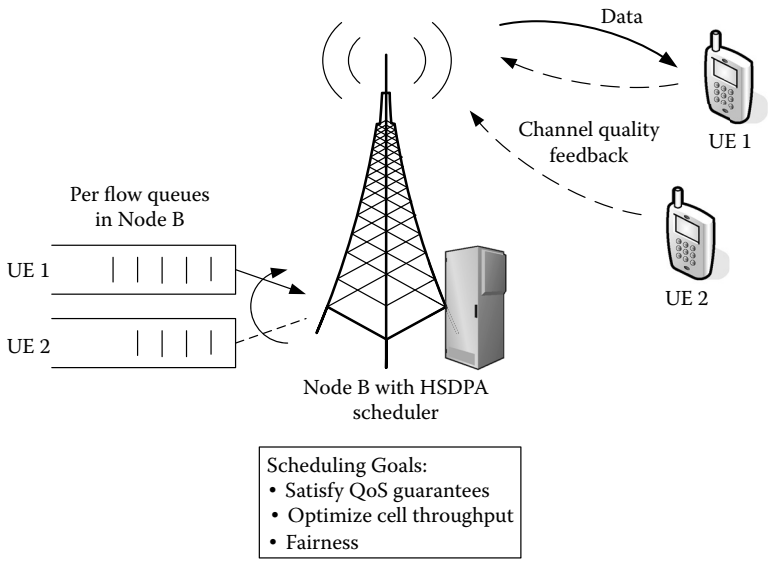

Figure 5.9 General goals of QoS-aware packet scheduling algorithms.

requirements. Thus, as shown in Figure 5.9, packet scheduling should aim to fulfill these QoS requirements instead of just focusing on cell throughput and fairness.

QoS classes and parameters have already been defined by 3GPP standards [2]. However, these standards do not define the scheduler implementation details, and thus provide flexibility to implementers and Node B vendors to choose their own scheduler in order to map these QoS classes and parameters.

Several schedulers that can offer QoS assurances in HSDPA have been proposed and studied in the literature [13]. Pedersen et al. [14] discuss different options for providing QoS using different QoS-aware schedulers. This section discusses various QoS schedulers. For quantitative analysis, different simulations are performed with a focus on video streaming services. The video traffic is generated by taking a video of 1-minute duration. The uncompressed video is a raw YUV video formed by concatenating three well-known Quarter Common Intermediate Format (QCIF) reference video sequences ("Foreman," "Mother and Daughter," and "News") in order to have a video of 1 minute and also to have a video with variable complexity. The video is then compressed using x264 codec [15] with H.264 baseline profile level 1.2 with no Bi-directional (B) frames, 15 frames per second, and strict 100-kbps CBR with a maximum 10% of bit rate variations. The EvalVid [16] methodology is used to perform EURANE simulations with the

packet traces. In the following text, some QOS schedulers are discussed. Please refer to Table 5.2 while reading the discussion.

5.3.2.1 Weighted Delay-Based Schedulers

A scheduler known as Modified Largest Weighted Delay First (MLWDF) is proposed in [17]. MLWDF chooses the user $i^* = \arg\max_i\{a_i w_i(t)\frac{R_i(t)}{\lambda_i(t)}\}$, where $w_i(t)$ can be the delay of head-of-line packet in the user queue, user queue length, or $\frac{Q_i(t)}{GBR_i}$, where $Q_i(t)$ is the number of tokens corresponding to the user token bucket algorithm, which is explained later. This choice ragarding the value of $w_i(t)$ depends on different versions of MLWDF and whether QoS guarantee is in terms of minimum delay or guaranteed bit rate (GBR_i) for user i.

MLWDF can provide two notions of QoS. First, it can achieve QoS requirements of the form $\Pr\{w_i(t) > D_i\} \leq \delta_i$, where $w_i(t)$ is the delay of the head-of-line packet in the user queue, D_i is the target delay, and δ_i is the probability of QoS requirement violation. Second, it should be noted that this QoS requirement in terms of delay can easily be rewritten in another form to provide minimum rate guarantees. Moreover, MLWDF can support mixed types of QoS guarantees in the system such that some of them are provided delay guarantees and others are provided rate guarantees. In the case when a minimum rate guarantee is provided, MLWDF uses a token

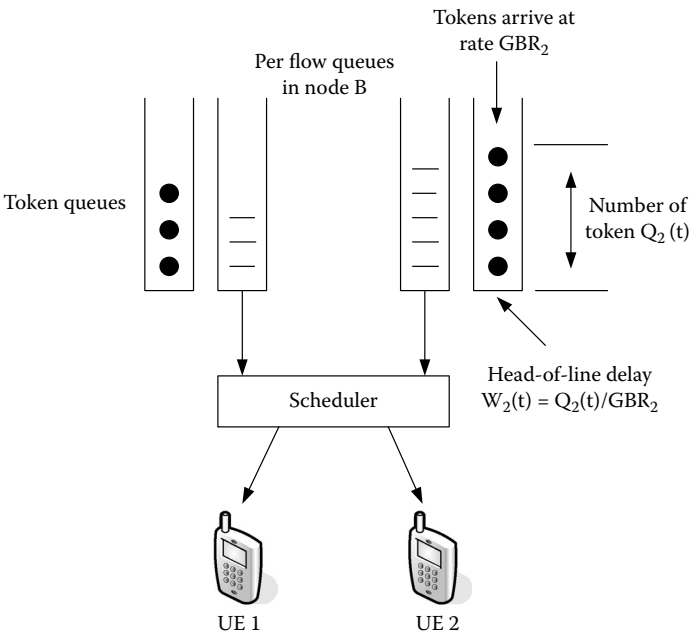

Figure 5.10 Using token queues with the MLWDF scheduling algorithm.

bucket control algorithm, as shown in Figure 5.10, where for each user, i tokens arrive at a constant rate equal to GBR_i. The MLWDF rule is used to schedule the users, and $w_i(t)$ is replaced by the delay of the head-of-line token in the token queue that, in turn, is equal to $\frac{Q_i(t)}{GBR_i}$ because tokens arrive at constant rate GBR_i. When a user is scheduled in the current TTI, a number of tokens equal to the amount of data sent for that user are removed from his token bucket.

The parameter a_i is an implementation parameter that reflects the strictness of QoS guarantees and in case of delay guarantees; [18] suggests the value of a_i to be $-log(\delta_i)/D_i$. In the case of rate guarantees, a_i is *not* equal to $-log(\delta_i)/GBR_i$ because GBR_i is not exactly equivalent to D_i. Any specific value of a_i, when rate guarantees are provided, is not suggested in [18], although it is recommended to set a higher value of a_i to provide stricter assurance of minimum rate guarantee for a given user i.

Figure 5.11 shows the benefits of using a QoS-aware scheduler, MLWDF, over Best-Effort schedulers when some users in the cell require QoS guarantees. A mixed scenario is considered in the simulation with some video streaming users and FTP users in the background. To decide on the maximum number of video streaming users that can be supported, a criterion is used such that 90% of the video streaming users should be satisfied across different simulation runs. The value of GBR_i is set to 110 kbps. This is slightly higher than the compressed video bit rate of 100 kbps. The value of a_i is set to 1.0 for all users. Figure 5.11 shows the percentage of satisfied users with respect to an increasing number of video streaming users. It should be noted that, in this figure, a video streaming user is assumed to be satisfied if PSNR (peak-signal-to-noise ratio) of the received video as compared to original uncompressed video is higher than 30 dB. The PSNR is calculated by considering the whole video as one image and then comparing it with the original uncompressed video. It should be noted that deterioration in video quality is caused due to lost packets and delayed packets that are discarded.

In Figure 5.11(a), it can be seen that in case of no Best-Effort flows, MLWDF can support a significantly greater number of users as compared to Best-Effort schedulers. In the case of Best-Effort load, generated using ten FTP flows, the difference in terms of the maximum number of video streaming users supported as compared to that of Best-Effort schedulers is even higher as shown in Figure 5.11(b). In fact, no Best-Effort scheduler is able to satisfy 90% of users in any case when ten FTP users are present in background, as shown in Figure 5.11(b). This is because the Best-Effort scheduler cannot guarantee minimum throughput, and this leads to packet losses that, in turn, degrade the video quality. On the other hand, a QoS-aware scheduler can provide a minimum rate guarantee for up to 18 video streaming users in the case of no Best-Effort load and 16 video streaming users in case of Best-Effort load.

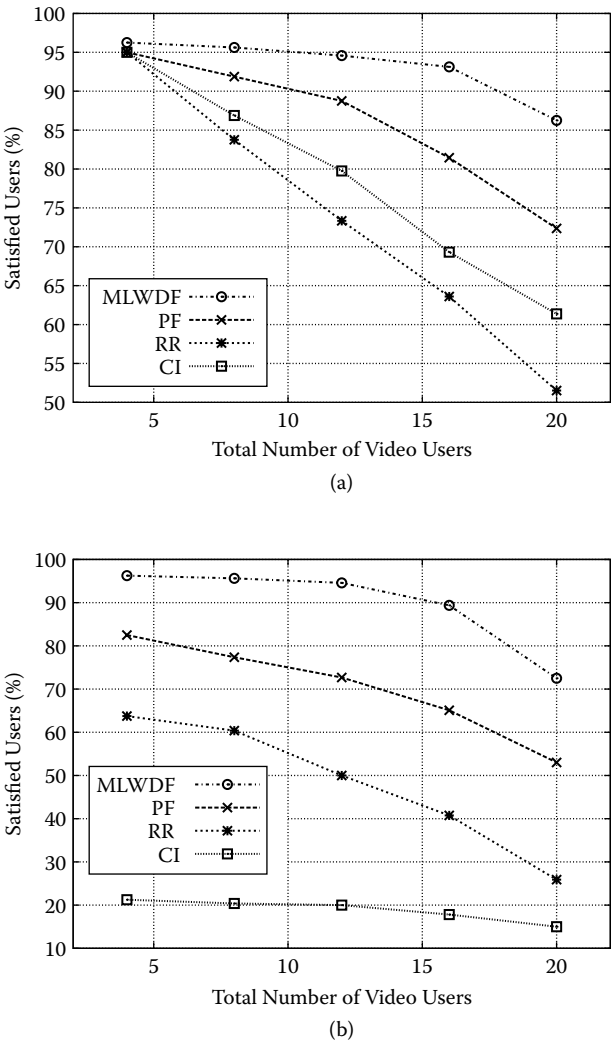

Figure 5.11 Percentage of satisfied video users with increasing number of video users. (a) No background traffic; and (b) ten FTP users as background traffic. A video user is assumed to be satisfied if video PSNR $>$ 30. Drop Tail queue management is used at RNC with 60 packets of queue limit. Discard timer is not set, and a credit-based flow control is used between RNC and Node B.

While studying MLWDF, another interesting scheduling function was reported in [18] and called the Exponential Rule (EXP). The EXP scheduler chooses the user

$$i^* = \arg\max_i \left\{ a_i \frac{R_i(t)}{\lambda_i(t)} \exp\left(\frac{a_i w_i(t) - \frac{1}{N}\sum_i a_i w_i(t)}{b + \sqrt{\frac{1}{N}\sum_i a_i w_i(t)}} \right) \right\}$$

where N is the total number of users and b is another implementation parameter. This function was further studied in [19]. The idea of the EXP scheduler is to keep the values of $a_i w_i(t)$, for different users, close to each other. This is because when the value of $a_i w_i(t)$ for user i is more than $\frac{1}{N}\sum_i a_i w_i(t)$ by a value that is of the order of $\sqrt{\frac{1}{N}\sum_i a_i w_i(t)}$, then the priority of that user starts to increase [18]. This in turn increases the probability of that user being scheduled. It was also noted in [18] that the term $\frac{1}{N}\sum_i a_i w_i(t)$ can be removed from the numerator in the EXP scheduler equation because it is common for all the users and is kept just for the purpose of illustration. Thus, only the term $\sqrt{\frac{1}{N}\sum_i a_i w_i(t)}$ present in the denominator matters in the scheduling decision.

Intuitively, it can be remarked that there are two differences between EXP and MLWDF. The exponential weight in EXP scheduler always remains more than 1.0, with $a_i \geq 0$ and $w_i(t) \geq 0$, after the common term $1/\mathrm{N}\sum_i \mathrm{a_i w_i(t)}$ is removed from the numerator in the EXP scheduler equation. Whereas in the case of MLWDF, the weight is equal to $w_i(t)$, which can take values close to or equal to 0 and, in this case, a user will not be scheduled even with a very high value of $R_i(t)$. Moreover, instead of absolute deterioration of QoS guarantees, EXP reacts to relative deterioration of QoS guarantees. The latter is because EXP tries to bring the values of $a_i w_i(t)$ for all users close to $\frac{1}{N}\sum_i a_i w_i(t)$, which in turn means that if QoS is bad for some users then it will become bad for all users. These two schedulers are further compared in the later sections.

Another point that can be made about both schedulers is that the scheduler function is not defined for Best-Effort users as all users get either delay or minimum throughput guarantees. Thus, when RNC does not set the values for GBR_i and Discard Timer, the behavior of MLWDF and EXP implementation is not defined. However, this can be easily overcome. For example, if RNC sets $GBR_i = 0$ for some flows, the scheduler implementation in Node B can set GBR_i to a small value, that is, more than 0 (for example, equal to 10 kbps). This means that the Best-Effort flows, even though they may no longer be called Best-Effort, will get at least 10 kbps of throughput. They can always get more throughput, depending on the channel conditions, as any additional capacity will be divided among all the users. Further it should be noted that even non-real-time flows can

always be assigned some QoS guarantees, depending on the QoS mapping defined by the operator.

5.3.2.2 Utility-Based Schedulers

As we discuss more QoS schedulers, it will be seen that most of these schedulers take the slot assignment decision and pick the user i^* such that

$$i^* = \arg\max_i \left\{ B_i(t) \frac{R_i(t)}{\lambda_i(t)} \right\} \tag{5.2}$$

where $B_i(t)$ represents a function that increases or decreases a user's priority, depending on his QoS. Note that a user's priority in turn affects his chances of being scheduled in the current TTI. In the case of MLWDF, $B_i(t) = a_i w_i(t)$; and in the case of EXP,

$$B_i(t) = a_i \exp\left(\frac{a_i w_i(t) - \frac{1}{N}\sum_i a_i w_i(t)}{b + \sqrt{\frac{1}{N}\sum_i a_i w_i(t)}} \right)$$

To design QoS schedulers, it was shown in [20] that, given a user utility $U_i(\lambda_i)$, it is possible to obtain a "good" scheduler that picks the user i^* such that

$$i^* = \arg\max_i \left\{ R_i(t) \frac{\partial U_i(\lambda_i)}{\partial \lambda_i} \right\} \tag{5.3}$$

Note in the above formulation that the corresponding utility functions for RR, Max C/I, and PF scheduling are $U(\lambda) = 1$, $U(\lambda) = \lambda$, and $U(\lambda) = \log\lambda$, respectively. However, it should be remarked that defining the required utility function is a difficult task. A Rate-Guarantee (RG) scheduler is designed in [21] using Equation (5.3). A utility function of the form $U(\lambda) = \log\lambda + 1 - \exp(-\beta \cdot (\lambda - GBR))$ is assumed, with $\beta > 0$, resulting in the RG scheduler with

$$B_i(t) = \begin{cases} 1 + \lambda_i(t) \cdot \beta \cdot \exp(-\beta \cdot (\lambda_i(t) - GBR_i)) & \forall i \in \mathcal{Q} \\ 1 & \forall i \in \mathcal{B} \end{cases} \tag{5.4}$$

which corresponds to the case where the minimum throughput, GBR_i, is required by a QoS user i; $\mathcal{Q}$ and $\mathcal{B}$ denote the set of QoS and Best-Effort (BE) users, respectively. Note that the exponential function ensures that whenever a user with minimum throughput guarantee GBR_i is getting a rate lower than GBR_i, then the exponential function will increase rapidly, which in turn will increase the chances of that user getting scheduled. On the other hand, when the user throughput is greater than GBR_i, then the value of the exponential function will decrease slowly. The implementation parameter β controls the aggressiveness of the exponential function, to

increase and decrease, depending on the value of the difference between the current user throughput and GBR_i.

Similarly, other schedulers based on delay guarantees can be designed [20]. In the literature, there exist several studies [14,20,22] that use RG to provide rate guarantees to QoS users. Unlike MLWDF and EXP, the RG scheduler has an option to schedule Best-Effort users that, in turn, are the users without any QoS guarantees. With RG, QoS users are scheduled according to their QoS requirements and any remaining throughput capacity is distributed in a proportionally fair manner to Best-Effort users. Whether a user flow is assigned to the Best-Effort class is decided by RNC that, in turn, sets the QoS parameters like *GBR* to 0 for those flows.

It should be noted that when the minimum throughput guarantee GBR_i of a QoS user is getting satisfied, and thus $\lambda_i(t) = GBR_i$, then $B_i(t) = 1 + \beta \cdot GBR_i$, which is a value that is higher than $B_i(t) = 1$ for Best-Effort users. The study in [22] thus replaces the term $\lambda_i(t) \cdot \beta$, before the exponential term in Equation (5.4), with α, for QoS users, and replaces $B_i(t) = 1$ with $B_i(t) = 1 + \alpha$ for Best-Effort users. This implies that the value of $B_i(t)$ for Best-Effort users becomes the same as that for QoS users when their QoS guarantees are being satisfied. However, this variant is not recommended for implementation, as it was reported in [23] that it shows disparity among different QoS users when they have different values of GBR_i, as discussed below.

The work in [23,24] studied the RG scheduler and it was found that both the original RG scheduler in [20] and its variant in [22] suffer from the deterioration of rate guarantees of the QoS users as the number of "active" BE users $n_{be} = |\mathcal{B}|$ increases. This is because as n_{be} increases, the value of λ_i for BE users decreases. This in turn increases the term $B_i(t)/\lambda_i$ in Equation (5.2) for BE users as $B_i(t) = 1$ remains constant. This means that BE users will start taking more resources from QoS users as their number increases. Moreover, the above schedulers, especially the variant in [22], are reported to be biased toward some users, depending on their values of rate guarantees. To solve these issues, the work in [24] proposes an alternate scheduler called Required Activity Detection (RAD), which is not a utility-based scheduler and is discussed in the next section, and the work in [23] proposes a normalized version of the RG scheduler called Normalized Rate Guarantee (NRG).

NRG is another utility-based scheduler that is based on RG. It overcomes the drawbacks of RG and is a normalized version of the RG scheduler. Unlike RG, which uses the absolute difference between λ and *GBR*, NRG uses a normalized term $(\lambda - GBR)/GBR$ in order to increase or decrease the exponential function based on the percentage drop in the throughput rather than using the absolute difference. Another modification is done to ensure that the increase in BE load should not deteriorate the quality of the QoS flows, and the increased BE load should be shared among BE users

only. Thus, the goal is to segregate BE users from QoS users, as explained below.

NRG assumes that the utility for the QoS users is such that $U_{\mathcal{Q}}(\lambda) = GBR \cdot (\log(\lambda)+1-\exp(-\beta \cdot \frac{\lambda-GBR}{GBR}))$. For Best-Effort users, NRG considers, similarly to the RG scheduler, that the rate guarantees are "always satisfied," and the utility for BE users is such that $U_{\mathcal{B}}(\lambda) = \frac{k_{be}}{n_{be}} \log(\lambda)$. The value of k_{be}, which is an implementation parameter, determines the proportion of resources allocated to the BE users and the n_{be} term, similar to the concept used in [24], in the denominator, will ensure that if the number of BE users increases, then it will not increase the load on the resources of QoS users. Furthermore, a value $n_{be}^{(min)}$ can be chosen such that $B_i(t) = k_{be}/n_{be}^{(min)}$ if $n_{be} < n_{be}^{(min)}$ to avoid the division by zero and to slightly improve the QoS when the number of BE users goes low.

Using the above utilities and Equation (5.3) results in the NRG scheduler with

$$B_i(t) = \begin{cases} GBR_i + \lambda_i(t)\beta \cdot \exp\left(-\beta \cdot \frac{(\lambda_i(t)-GBR_i)}{GBR_i}\right) & \forall i \in \mathcal{Q}, \\ \frac{k_{be}}{n_{be}} & \forall i \in \mathcal{B} \end{cases} \tag{5.5}$$

The parameter β determines the aggressiveness of the scheduler to the percentage drop in the user throughput and $\beta = 6.0$ is used in [23]. The typical values of k_{be} considered in this chapter are between 550 and 1500 when GBR_i and $\lambda_i(t)$ are in kbps unit.

It should be noted that the term $B_i(t)/\lambda_i(t)$ in the scheduler Equation (5.2), with $B_i(t)$ defined by NRG in Equation (5.5), becomes equal for all the QoS users when they achieve their minimum throughput or when the percentage drop in their throughput is the same. Thus, it is called a Normalized Rate Guarantee scheduler because both the terms $(\lambda_i(t)-GBR_i)/GBR_i$ for QoS users and $B_i(t)$ for BE users are normalized. The results reported in [23] show that NRG is able to avoid the deterioration in rate guarantees of the QoS users when the number of Best-Effort users are increased in the system; whereas the increase in Best-Effort load significantly deteriorates the QoS when RG is used. In addition, NRG was shown [23] to be unbiased toward different values of GBR_i.

Figure 5.12 compares the performance of NRG with the performance of a Proportionally Fair (PF) scheduler, a Best-Effort scheduler. Note that the goal of a QoS scheduler is to, first, satisfy the QoS guarantees and then maximize the cell throughput. Thus, we focus on the Best-Effort throughput with respect to the number of QoS users in the system. The setting is similar to that used in a previous section ("Weighted Delay-Based Schedulers") except that Drop Tail is not used and instead a relaxed Discard Timer value of 5.0 s is used. Moreover, a user is assumed to be satisfied if the amount of discarded packets due to Discard Timer is less than 5%.

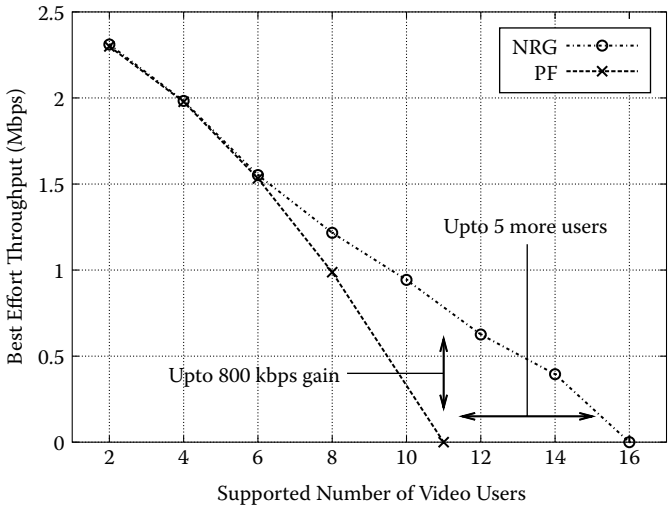

Figure 5.12 QoS-aware scheduler results in the gains in terms of overall throughput and maximum number of QoS users supported in the system. In the simulations, a relaxed Discard Timer value of 5.0 s is used instead of using Drop Tail. The credit-based rate control algorithm between RNC and Node B is not used.

In the case of NRG, ten FTP users are used as Best-Effort users, and the number of video users is increased gradually until 90% of the video users remain satisfied across 20 simulation runs and overall throughput obtained by Best-Effort users is noted. Slight optimization is used with NRG such that when the number of video users is less than ten, then the value of k_{be} in the NRG scheduler function is taken to be 1000 with λ and *GBR* in kbps. But, when the number of video users is more than ten, the value of k_{be} decreased to 500. On the other hand, in the case of PF, the number of video users as well as the number of Best-Effort users varies. It should be noted that for a given number of video users and in the case of the PF scheduler, when the Best-Effort load is increased gradually, it reaches a point where just 90% of the video users are satisfied across 20 simulation runs. This throughput value is noted, for PF, with respect to the number of video users, as it is the maximum Best-Effort throughput possible that can still ensure 90% video user satisfaction.

From Figure 5.12 it is clear that NRG outperforms PF when the number of video users in the system increases. The gain is in terms of up to 800 kbps of additional overall throughput for Best-Effort flows and up to five more video users that can be supported in the system. The gain over PF is due to the fact that after some point, the PF scheduler cannot accept more Best-Effort users as it would not be able to satisfy the QoS users. But NRG can accept any amount of Best-Effort load because it first satisfies the QoS

Table 5.2 QoS Scheduling Algorithms

Scheduler	*QoS Guarantee*	*Choose User* $i^* = \arg\max_i \left\{ \frac{B_i(t)R_i(t)}{\lambda_i(t)} \right\}$	
		$B_i(t)$ *for QoS Users*	$B_i(t)$ *for BE Users*
MLWDF	Delay/Rate	$a_i w_i(t)$	Not defined
EXP	Delay/Rate	$a_i \exp\left(\frac{a_i w_i(t) - \frac{1}{N}\sum_i a_i w_i(t)}{b+\sqrt{\frac{1}{N}\sum_i a_i w_i(t)}}\right)$	Not defined
RG	Rate	$1 + \lambda_i(t)\beta \exp\left(-\beta(\lambda_i(t) - GBR_i)\right)$	1
RAD	Rate	$\begin{cases} \frac{GBR_i}{\lambda_{sch}^{(i)}(t)} & \text{if } \rho(t) < 1, \\ \frac{GBR_i/\lambda_{sch}^{(i)}(t)}{\rho(t)} & \text{if } \rho(t) \geq 1 \end{cases}$ where $\rho(t) = \sum_i \frac{GBR_i}{\lambda_{sch}^{(i)}(t)}$	$\begin{cases} \frac{1-\rho(t)}{n_{be}} & \text{if } \rho(t) < 1, \\ 0 & \text{if } \rho(t) \geq 1 \end{cases}$ where $\rho(t) = \sum_i \frac{GBR_i}{\lambda_{sch}^{(i)}(t)}$
NRG	Rate	$GBR_i + \lambda_i(t)\beta \exp\left(\frac{-\beta(\lambda_i(t) - GBR_i)}{GBR_i}\right)$	$\frac{k_{be}}{n_{be}}$

Note: Here, for a given user i at time t, $\lambda_i(t)$ is the exponentially weighted moving average throughput, $R_i(t)$ is the instantaneous supportable data rate, and $w_i(t)$ is either head-of-line packet delay or {Number of Tokens}/GBR_i as shown in Figure 5.10. $\lambda_{sch}^{(i)}$ is the effective scheduling throughput based on only the time slots when user i is actually scheduled. N and n_{be} are the total number of users and Best-Effort users, respectively. The parameters a_i, b, β, and k_{be} are implementation parameters.

users and then divides the remaining capacity, if any, among the Best-Effort users, which is the goal of NRG.

5.3.2.3 Required Activity Detection Scheduler

The idea of a RAD scheduler is that when user i is being served at a data rate of $\lambda_{scb}^{(i)}(t)$ in the time slot when that user is scheduled, then that user needs to be scheduled for $GBR_i/\lambda_{scb}^{(i)}(t)$ time slots per unit time. This term is called the required activity of the user. Thus, this term is taken to be $B_i(t)$ in Equation (5.2) because then it acts as weighted PF, and over the long term it tends to divide the total time slots among different users in proportion to $B_i(t)$. This, is turn, ensures a minimum bit rate of GBR_i for user i, as shown by the results in [24].

RAD is a QoS-aware scheduler with different definitions of $B_i(t)$ during congestion and during no congestion. Congestion is defined with respect to the total load $\rho(t) = \sum_i \frac{GBR_i}{\lambda_{scb}^{(i)}(t)}$. When $\rho(t) < 1$, the system is not congested as there are resources available that can satisfy QoS guarantees at that time.

In this case, $B_i(t)$ is taken as

$$B_i(t) = \begin{cases} \dfrac{GBR_i}{\lambda_{scb}^{(i)}(t)} & \forall i \in \mathcal{Q}, \\ \dfrac{1-\rho(t)}{n_{be}} & \forall i \in \mathcal{B} \end{cases} \tag{5.6}$$

On the other hand, when the system is congested and $\rho(t) \geq 1$, then

$$B_i(t) = \begin{cases} \dfrac{GBR_i/\lambda_{scb}^{(i)}(t)}{\rho(t)} & \forall i \in \mathcal{Q}, \\ 0 & \forall i \in \mathcal{B} \end{cases} \tag{5.7}$$

where n_{be} is the number of Best-Effort users and $\lambda_{scb}^{(i)}(t)$ is the effective scheduling throughput based only on TTIs when user i is actually scheduled. It is calculated as follows:

$$\lambda_{scb}^{(i)}(t) = \left(1 - \frac{s_i}{\tau_{scb}}\right)\lambda_{scb}^{(i)}(t-\Delta t) + \frac{s_i}{\tau_{scb}} R_i(t)$$
$$\text{with } s_i = 1$$

when user i is scheduled in the current time slot, and $s_i = 0$ otherwise. It should be noted that $\lambda_{scb}^{(i)}(t)$ is computed in a slightly different way as compared to $\lambda_i(t)$ in Equation (5.1) because of the different value of τ_{scb} and; in addition, when user i is not scheduled in the current time slot, unlike $\lambda_i(t)$ that decreases, $\lambda_{scb}^{(i)}(t)$ is exactly set to the previous value $\lambda_{scb}^{(i)}(t-\Delta t)$. The performance of the RAD scheduler is studied in the next section.

5.3.2.4 Comparison among the Different QoS-Aware Schedulers

Before comparing different QoS aware schedulers, it should be noted that a fair comparison is very difficult because of numerous implementation parameters and, more importantly, due to different configuration options offered by the different QoS-aware schedulers. In fact, we can actually compare the advantages of different QoS-aware schedulers discussed in this chapter with respect to the different configuration options that are offered. The advantage of MLWDF and EXP is that they can support delay-based as well as rate-based QoS guarantees, simultaneously. The positive point about utility-based schedulers is that if one can design the utility of a QoS service, then one can design the corresponding optimal scheduler. Regarding the RAD scheduler, its advantage lies in the fact that unlike other QoS-aware schedulers that require the tuning of numerous implementation parameters, it does not have any implementation parameters to tune and only requires the value of *GBR*. In addition, unlike MLWDF and EXP, the NRG and RAD schedulers have the option to first satisfy the QoS guarantees

and then divide the remaining capacity among the Best-Effort users, such as those with $GBR = 0$, without affecting the QoS users.

To further compare different QoS-aware schedulers, two types of settings are considered. Because the focus of this chapter is on downlink-centric services such as video streaming, Video-on-Demand, FTP, and others, we assume that these services will have a varied range of delay requirements. Thus, RNC can set different values of the Discard Timer, such as a strict timer value of 1.5 s or lower and up to much more relaxed values like 5.0 s or no Discard Timer at all. This section thus compares the performance of different schedulers in two scenarios: one in which the Discard Timer value is set to 1.5 s and the other in which it is set to 5.0 s. We look at the percentage of packets belonging to the QoS users that are dropped by the Discard Timer. To provide a fair comparison of these schedulers, their numerous implementation parameters are tuned, after a lot of simulation runs, such that the Best-Effort throughput becomes equal to that of other schedulers under the same conditions. Most of the time, these implementation parameters are re-tuned whenever the configuration changes, for example, when the number of video users is changed. Similar to previous sections, the results reported in this section are the average values obtained after 20 independent simulation runs. In case of MLWDF and EXP schedulers that do not define the function for Best-Effort users, a low rate guarantee of 10 kbps is used and parameter a_i is moved to match the required overall Best-Effort throughput. It is observed that a low rate guarantee for Best-Effort users results in better overall throughput as compared to when a higher rate guarantee, such as 28 kbps [19] or 64 kbps, is used with MLWDF and EXP for Best-Effort users. In the case of NRG, the value of β is taken as 6.0 and the parameter k_{be} is used for tuning. Because there are no parameters that can be tuned in the case of RAD, the value of *GBR* is moved such that for a video of 100 kbps, the values of *GBR* are chosen that are increasingly higher than 100 kbps. Other simulation parameters are the same as those in previous sections.

It can be seen in Figure 5.13 and Figure 5.14 that in the case of MLWDF, it is not possible to match the Best-Effort throughput with other QoS schedulers without significantly increasing the discard rate of QoS users. This is a disadvantage of MLWDF, which uses a weighted delay approach, whereas other QoS-aware schedulers divide the remaining capacity using a Proportional Fair-like algorithm. This is true even for the EXP scheduler, and the difference derives from the weighted delay approach of MLWDF that varies the value of $B_i(t)$ from 0 to some value that in turn performs poorer as compared to PF-like schedulers in terms of overall throughput. In contrast to MLWDF, the lowest possible value of the exponential function in EXP is 1, which in turn means a_i for $B_i(t)$. Thus, from Figures 5.13 and 5.14, it can be seen that MLWDF results in the loss of up to 400 kbps of overall throughput for similar and sometimes a worse percentage of discarded packets.

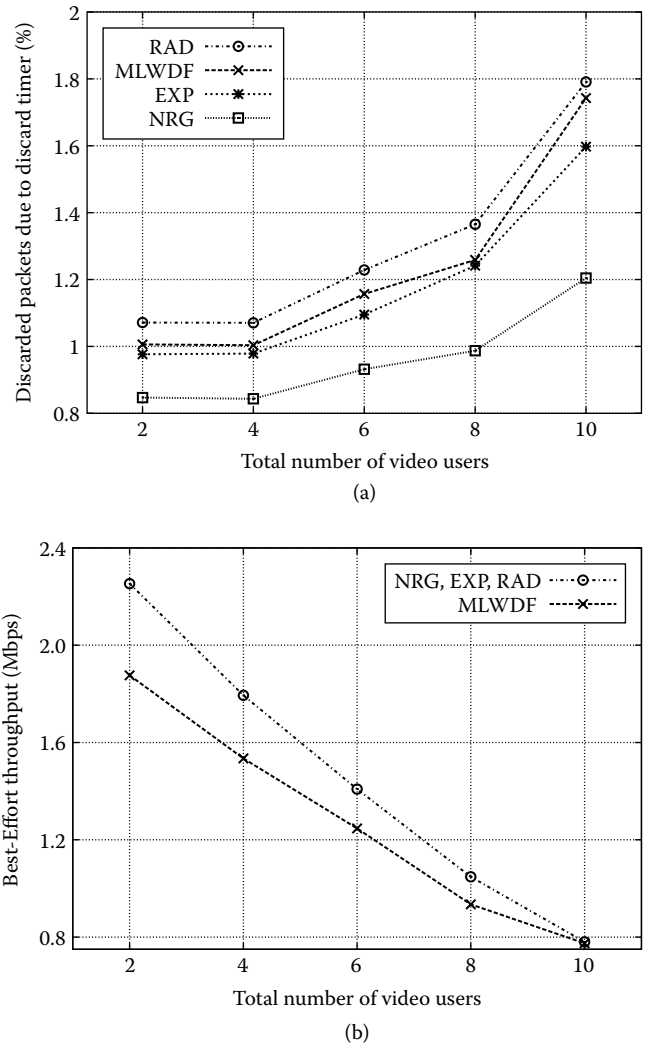

Figure 5.13 Comparison of different QoS schedulers when Discard Timer value is 1.5 s. All schedulers are tuned such that the BE throughput is the same with the exception of MLWDF. For MLWDF, higher throughput is not possible without significantly increasing the average loss rate. (a) Average loss rate of video flows and (b) Best-Effort throughput.

Another point to observe is that when the Discard Timer value is strict and equal to 1.5 s, the NRG performs best, as it results in the lowest percentage of discarded packets with the same, or better in the case of MLWDF, overall throughput. This is because the exponential function used in NRG behaves aggressively toward the drops in throughput values for QoS users;

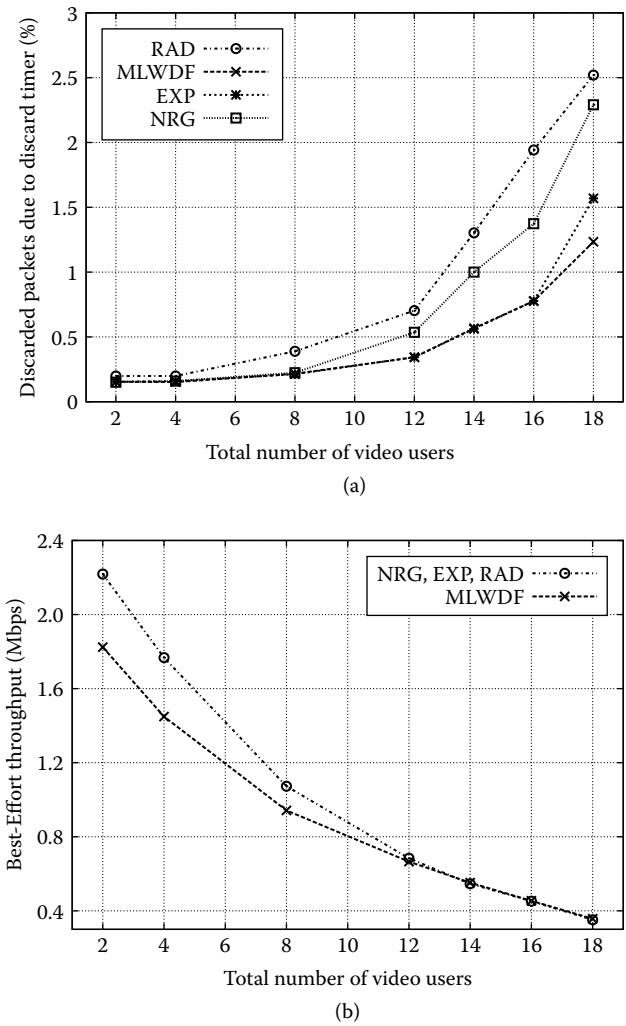

Figure 5.14 Comparison of different QoS schedulers with relatively relaxed Discard Timer value of 5.0 s: (a) Average loss rate of video flows and (b) Best-Effort throughput.

whereas, other schedulers have a relatively more relaxed approach when throughput drops beyond *GBR*. On the other hand, when a relaxed Discard Timer value is used, such as 5.0 s, then the EXP scheduler performs best with the exception of MLWDF performing slightly better when the number of video users is close to the maximum number of video users that can ever be supported. The RAD scheduler performs slightly worse than other schedulers because it has the most relaxed approach to satisfy

throughput guarantees and, in addition, it starves the Best-Effort users when QoS guarantees are not getting satisfied, which means slightly lower throughput for Best-Effort users. This in turn increases the discard rate when the *GBR* value is tuned to match the Best-Effort throughput of other schedulers, even when *GBR* is always kept above 110 kbps.

In addition, it is found that although MLWDF has a slight disadvantage in terms of lower overall throughput, it still supports the most or an equal number of QoS users, when compared with other schedulers, in all the cases when Best-Effort traffic is not present. This is because when the system is 100% loaded by QoS users, the Best-Effort throughput is minimal anyway and does not make any difference. It is observed that for MLWDF, NRG, EXP, and RAD, the maximum number of video users supported in two cases are 12, 12, 9, and 9 when the Discard Timer is 1.5 s; and 18, 16, 17, and 13 when the Discard Timer is 5.0 s, respectively. This special case, of maximum number of QoS users supported, corresponds to the case when no background traffic is present.

5.4 Discussion

This chapter considered some of the existing scheduling algorithms with a focus on downlink-centric services. Services such as VoIP that are equally uplink centric were not discussed. Moreover, it was assumed that only one user is scheduled in one TTI, and code multiplexing, which allows multiple users to be scheduled in same TTI, was not considered. It should be noted that the use of code multiplexing can increase the intra-cell interference and thus it is not always relevant in the case of downlink-centric services. However, in the case of services such as VoIP, it becomes essential to use code multiplexing. This is because of the inherent delay characteristics of VoIP data flow, which means end-to-end delay requirements on the order of 200 ms and data rates as low as 16 kbps. So, at any given time, it is highly likely that the amount of data available for transmission will be less than that which can be sent in one TTI. Thus, most of the time, with VoIP, there will be additional resources available for more than one user in each TTI. The absence of code multiplexing can mean inefficient utilization of radio resources, and thus code multiplexing is highly recommended in the case of VoIP service. When considering code multiplexing and other enhancements, a total capacity of up to 120 VoIP users was reported in [25] over HSPA with 3GPP Release 7.

The role of the scheduling priority indicator (SPI) was not discussed in analyzing the different schedulers. One simple approach is to schedule users with strict priority, depending on SPI, such that lower priority users are not scheduled until there is data available in the queues of

higher priority users. However, this approach will lead to frequent starvation of lower priority users. In this chapter, a "soft priority" approach was used while scheduling users such that lower priority users, for example Best-Effort users, are not completely segregated from QoS users. Even when some backlog data is available for QoS users, Best-Effort users can still be scheduled, depending on the scheduler algorithm and the current channel conditions. In the context of the schedulers discussed in this chapter, the value of SPI will be used to determine the implementation parameters of the respective schedulers.

Moreover, it should be noted that this chapter focused mainly on scheduling algorithms present in Node B. Some other resource management mechanisms (such as call admission control, buffer management algorithms at RNC, congestion control, handover management, and QoS mapping), although outside the scope of this chapter, are very important for overall QoS satisfaction of the users. These mechanisms, combined with the scheduling algorithm, will determine the overall impact on user-perceived quality-of-experience (QoE).

While assessing the impact of several schedulers on the QoS satisfaction of the users, we used objective metrics such as PSNR and loss rate. However, it is well known that objective metrics such as these do not necessarily correlate well with user-perceived quality when a network is involved. See, for example, [26]. On the other hand, subjective evaluation of video quality is very costly, and one good way to obtain a quality metric as correlated as possible with human visual perception is to use some recently proposed techniques such as those proposed in [27–29].

5.5 Conclusion

This chapter focused on HSDPA scheduling principles and different scheduling algorithms. The key point of HSDPA is its fast and adaptive packet scheduling. This chapter discussed some existing HSDPA packet schedulers. These schedulers were evaluated with a focus on download-centric services in a 3G network. For evaluating the performance, a mixed traffic scenario was considered.

In terms of the maximum number of users that can be served at one time, it was found that QoS-aware schedulers performed better. For the low-load conditions, Best-Effort schedulers could satisfy different users; but as the load increased, only QoS schedulers were able to guarantee a minimum quality to 90% of the video users. The trade-off was the lower per-user throughput that the BE users were getting in comparison. Nevertheless, the QoS-aware schedulers were still dividing the remaining capacity among the BE users in a fair manner.

References

[1] EURANE, http://www.ti-wmc.nl/eurane/.

[2] 3GPP TR 23.107 v 5.13.0, Technical Specification Group Services and System Aspect; Quality of Service (QoS) Concept and Architecture, Release 5, 2005.

[3] 3GPP TS 25.433 v 5.16.0, UTRAN Iub Interface Node B Application Part (NBAP) Signalling, October 2006.

[4] S. McCanne and S. Floyd, ns Network Simulator, http://www.isi.edu/nsnam/ns/.

[5] M. Gudmundson, Correlation model for shadow fading in mobile radio systems, *Electronics Lett*., 27(23): 2145–2146, 1991.

[6] A. Jalali, R. Padovani, and R. Pankaj, Data throughput of CDMA-HDR A high efficiency-high data rate personal communication wireless system, in *Proc. IEEE VTC 2000 Spring*, 3: 1663–1667, 2000.

[7] S. Borst, User-level performance of channel-aware scheduling algorithms in wireless data networks, *IEEE/ACM Trans. Networking*, 13(3): 636–647, 2005.

[8] T.E. Kolding, Link and system performance aspects of proportional fair scheduling in WCDMA/HSDPA, in *Proc. IEEE VTC 2003—Fall*, 3: 1717–1722, October 2003.

[9] F. Kelly, Charging and rate control for elastic traffic, *Eur. Trans. Telecommun*. 8: 33–37, 1997.

[10] 3GPP2, 1xEV-DV Evaluation Methodology (v14), C30-20030616-043R1, April 2003.

[11] T. Bonald, A score-based opportunistic scheduler for fading radio channels, in *European Wireless 2004*, Barcelona, Spain, February 2004.

[12] G. Barriac and J. Holtzman, Introducing delay sensitivity into the proportional fair algorithm for CDMA downlink scheduling, in *IEEE Seventh Int. Symp. Spread Spectrum Techniques and Applications, 2002*, September. 2002.

[13] P. Ameigeiras, Packet Scheduling and Quality of Service in HSDPA, Ph.D. dissertation, Institute of Electronic Systems, Aalborg University, Denmark, October 2003.

[14] K.I. Pedersen, P.E. Mogensen, and T.E. Kolding, QoS considerations for HSDPA and performance results for different services, in *Proc. IEEE VTC 2006-Fall*, September 2006.

[15] x264, http://www.videolan.org/developers/x264.html.

[16] J. Klaue, B. Rathke, and A. Wolisz, EvalVid—A framework for video transmission and quality evaluation, in *Proc. 13th Int. Conf. Modelling Techniques and Tools for Computer Performance Evaluation*, Urbana, IL, September 2003.

[17] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijayakumar, Providing quality of service over a shared wireless link, *IEEE Commun. Mag*., 39(2): 150–154, February 2001.

[18] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting, CDMA Data QoS Scheduling on the Forward Link with Variable Channel Condtions, *Bell Labs Technical Memo*., April 2000.

[19] S. Shakkottai and A. Stolyar, A Study of Scheduling Algorithms for a Mixture of Real- and Non-Real-Time Data in HDR, *Bell Labs Technical Memo*., August 2000.

[20] P. Hosein, QoS control for WCDMA high speed packet data, in *IEEE Proc. Int. Workshop on Mobile and Wireless Network Commun.*, September 2002, pp. 169–173.

[21] P. Hosein, A TCP-friendly congestion control algorithm for 1xEV-DV forward link packet data, in *The 57th IEEE Semiannual Vehicular Technol. Conf., 2003. VTC 2003—Spring*, April 2003, pp. 621–625.

[22] M. Lundevall, B. Olin, J. Olsson, N. Wiberg, S. Wanstedt, J. Eriksson, and F. Eng, Streaming applications over HSDPA in mixed service scenarios, in *Proc. IEEE VTC 2004–Fall*, 2: 841–845, September 2004.

[23] K.D. Singh and D.Ros, Normalized rate guarantee scheduler for high speed downlink packet access, in *IEEE GLOBECOM'07*, Washington, D.C., November 2007.

[24] T.E. Kolding, QoS-aware proportional fair packet scheduling with required activity detection, in *Proc. IEEE VTC 2006—Fall*, September 2006.

[25] H. Holma, M. Kuusela, E. Malkamaki, K. Ranta-aho, and C. Tao, VOIP over HSPA with 3GPP Release 7, in *IEEE 17th Int. Symp. on Personal, Indoor and Mobile Radio Commun.*, September 2006.

[26] R. Wu, T. Ferguson, and B. Qiu, Digital video quality using quantitative quality metrics, in *4th Int. Conf. Signal Processing*, Beijing, China, 1998, pp. 1013–1016.

[27] S. Mohamed and G. Rubino, A study of real-time packet video quality using random neural networks, *IEEE Trans. Circuits and Systems for Video Technol.*, 12(12): 1071–1083, December 2002.

[28] G. Rubino, M. Varela, and S. Mohamed, Performance evaluation of real-time speech through a packet network: A random neural networks-based approach, *Performance Evaluation*, 57(2): 141–162, May 2004.

[29] G. Rubino, P. Tirilly, and M. Varela, Evaluating users' satisfaction in packet networks using random neural networks, in *Proc. ICANN'06*, Athens, Greece, September 2006.